INTERPRETING HAND TREMOR WITH AN EXPERT SYSTEM TO ASSIST WITH STEERING A POWERED WHEELCHAIR

DA Sanders and A Gegov

School of Engineering, University of Portsmouth, Portsmouth, United Kingdom

david.sanders@port.ac.uk

Abstract.

Simple expert systems are presented that will allow more people to use powered wheelchairs. The systems interpret hand tremor and provide joystick position signals. Signals are mixed with ultrasonic sensor data to identify potentially hazardous situations and assist users to find a safe course. Results are discussed from a series of timed tasks completed by users using a joystick. They suggest that the amount of sensor support should be varied depending on circumstances and skill. Drivers completed progressively more complicated courses both with and with-out sensors and the most recently published systems are used to compare results. The new expert systems consistently out-performed the most recently published systems.

Keywords: powered-wheelchair, steering, expert, joystick, sensor, ultra-sonic.

Introduction

This paper describes a simple expert system for a powered-wheelchair that infers joystick position from users who may have shaky hands and then mixes that position data with data from ultrasonic sensors. The system can assist users in potentially hazardous situations and allow them to negotiate various terrains and obstacles. The system could be especially useful to provide independent mobility earlier for children.

Control systems for powered wheelchairs have tended to be open loop. Users have indicated a direction and the powered-wheelchair then moved in the required direction. Disturbances include differences in wheels or their different reaction to surfaces, and surface or gradient [1,2,3]. Powered-wheelchairs are generally guided using manual controls, often joysticks [4,5] although other devices are available, such as: switches [4], pointers [6,7] or custom built, such as Virtual Reality interfaces [8]. Users have usually been left to react to disturbances but the new system uses sensors to assist them.

Sensor systems

Powered-wheelchairs need to navigate around obstructions. Various sensors have been used to achieve that: light/laser [9], ultrasonic [10,11,12,13] and infra-red [14,15] . Positioning has used odometry, gyro, tilt and acoustic. GPS [16] is a de facto positioning system but GPS does not operate easily indoors. Vision opens up new possibilities [17,18,19,20] but vision requires more data processing and has been relatively ex-pensive and complicated [21,22]. Most wheelchairs rely on detection and guidance by a human being, sometimes using haptic force feedback [23].

Ultrasonic ranging was selected to assist because it was simple and robust. Recently published ultra-sonic sensor systems include [12,13]. They used 40 KHz ultrasonic transmitter and receiver pairs mounted in front of a powered-wheelchair. The system transmitted a 1ms pulse of ultrasonic energy and the pulse was reflected from objects in its path. Some reflected energy returned. Distance from sensors to object was then calculated from time taken for the pulse to return. With suitable processing the ultrasonic image was converted to a simple representation of the environment and objects in the powered-wheelchair path were detected.

In the new work described in this paper, the powered-wheelchair was initially con-trolled through a joystick. A controller interpreted joystick control signals and pro-vided power for the motors. The wheelchair was electrically powered with a front wheel drive chassis and fiberglass body. The base was a heavy steel plate chassis to provide stability and rigidity. Two driven wheels were at the front and two trailing casters at the back. Ultrasonic sensor pairs could be mounted over each driving wheel and in the middle at the front.

Trailing casters supported the rear and driving wheels were powered by two 12V DC motors through a worm drive right angle reduction gearbox. Correction was applied by means of differential motor drive [2]. Altering the differential of rotational speed of the driving wheels affected steering. The wheelchair consisted of a power source, motors, input device and a controller. Power, communications, joystick, interfaces and potentiometric and input devices are described in [12,13].

The direct link between the wheelchair and joystick was severed and a computer processed control information. Three modes of operation were possible in order to compare the performance of the new algorithms: Joystick data could be processed and sent to the controller without modification; or, sensors were activated and interrogated by the computer and the computer modified the wheelchair path using the most recently published methods; or, sensors were activated and interrogated by the computer and the computer was programmed to modify the powered-wheelchair path using new algorithms described in this paper.

New hierarchical code was constructed that was similar to levels described in [24,25]. Algorithms applied the following rules: (1) User remained in overall control; (2) Systems only modify trajectories when necessary, and (3) Movements were smooth and controlled. An imaginary potential field was generated around objects in response to sensor information [26,27] to assist users if the powered-wheelchair was approaching an object. The ultrasonic transmitters required a pulse of 3ms duration. If speed of sound in air is assumed to be 330m/s… physical length of a 3ms pulse of sound is 0.99m. Allowing for the pulse to leave the transmitter, bounce off an object and return to the receiver, then minimum range for a 3ms pulse would be 0.5m. Because closer ranges were required, shorter pulse lengths were needed. Pulse lengths of 10us, 100us, 500us and 1ms were examined. A range finder was created to automatically switch between pulse lengths as the range changed. If no object was detected, the range finder hunted by systematically increasing pulse length.

Ultrasonic sensors tended to be noisy and return misreads. A method for filtering out misreads was selected to improve sensor reliability that was based on Histogramic In-Motion Mapping. Volumes in front of each sensor were divided into a simple grid of three volumes: near, middle and far. They were stored as an array. When a range was returned, it was classified as near, middle or far.

Array elements represented an area where an object was detected. They were incremented by a higher number, for example, three. Other array elements were decremented by a lower number, for example, one. Arrays typically had a maximum value of 15 and a minimum of zero. This gave three simple three-element histogrammic representations of the environment. An object occupying a grid element would cause that element to quickly ramp in value to the maximum. Random misreads in the other elements incremented that element temporarily, but the value of false reads were decremented each time the system updated. If the object moved to a different element, the new element quickly ramped up to its maximum value and the old element ramped down to noise level. Reliable range could be acquired within 0.5s.

Interpreting the joystick

A standard Penny and Giles Potentiometric joystick was fitted that contained two potentiometers to provide two voltages. Joystick position could be read by an A/D converter as a set of Cartesian co-ordinates. That was not convenient as co-ordinates did not provide joystick signal direction or magnitude. Cartesian co-ordinates were converted to polar co-ordinates using trigonometrical functions and Pythagarus'. Joystick data was used in the form: $|J|\angle\theta$, where $|J|$ was magnitude (or how far the joystick had been pushed) and $\angle\theta$ was the angle of the joystick. Standard mathematical functions from C libraries calculated Cartesian to polar conversion.

The angle of the joystick introduced a directional element which could not be integrated. The joystick angular position was quantified so that intended direction could be estimated. This allowed algorithms to measure the length of time that a joystick had been held in a consistent direction and helped the new systems to identify the wishes of the user. Joystick angles were defined as:

Spin left	1.54 – 2.36 radians
Spin right	5.50 – 6.28 radians
Turn left	0.89 – 1.54 radians
Turn right	0.00 – 0.69 radians
Forward	0.69 – 0.89 radians
Reverse	2.36 – 5.50 radians
Stop	magnitude<16

Joystick magnitude was calculated using:

Magnitude = sqrt((JS0*JS0)+(JS1*JS1))

where JS0 and JS1 were the Cartesian co-ordinates with the origin centered on the joystick stop position. Magnitude and angle were then used to calculate the sector that the joystick was occupying. The position and confidence of the joystick could be expressed as an array. Each joystick sector contained two array values:

"Angle Confidence" (0 to 15) indicated certainty that joystick was in a sector.
"Magnitude" indicated joystick position (demanded powered-wheelchair speed).

Joystick output was integrated to provide a level of confidence in user intentions. A histogrammic representation was then used as a pseudo-integrator. If the joystick was held in a position, the array element relating to that position was incremented to raise its overall value. All other array elements could then be decremented to reduce their effect. The array element with the highest value was used as the latest and most confident joystick position. A joystick array element could quickly ramp in value to maximum. Random joystick action in the other elements incremented them temporarily, but values of false reads were decremented each time the system updated. If the joystick moved to a different element, the new element quickly ramped up to maximum and the old element ramped down to the noise level or zero. Joystick position was represented as a histogram where the highest histogram element represented the most likely direction for the user to be indicating as the desired direction.

A module called JSArray tested joystick position and angle, and indicated which sector the joystick was occupying. The appropriate element of the "angle confidence" (Aconf) was then increased by magnitude 40. All Aconf elements were then decreased in magnitude by 30 to decay the un-occupied elements. The occupied element was therefore subject to an increase of 10 in magnitude and all other elements were subject to a decrease in magnitude of 30. This allowed the histogram elements to decay rapidly and build in value more slowly. A joystick array element was able to increase to its maximum value of 225 in a minimum time of 0.5 seconds (approximately) and decay to zero in approximately 170ms. The ramping and delay weighting factors were determined experimentally by driving the powered-wheelchair with several different weighting factors in operation. The delay induced in the response of the powered-wheelchair by the weighting factors could be set to an individual user or task. Rules were intended as generative rules of behaviour; given some set of inputs then rules determined what the output should be [28].

Expert system

Some people were more naturally dextrous and could learn to drive in less time than others. When familiarisation was completed, a user could drive effectively.

There were two real time inputs; the input device (joystick) and sensors. A user indicated speed and direction and the sensor system gathered information about the environment. A module called Sensor Expert analysed sensor information and made a recommendation for a path to prevent collisions. Data often conflicted. Another expert, called Fuzzy Mixer considered both inputs and was responsible for motor controller outputs. Joystick Monitor was responsible for interpreting the wishes of the user. Variables such as joystick position and consistency were examined by Joystick Monitor to assess the desired trajectory.

Fuzzy Mixer apportioned control effort between joystick and sensor systems. It matched joystick and sensor recommendations, examined conflicts and kept controller voltage within parameters. It received information (or advice) from Sensor Expert, Joystick Monitor and Doorway. For safety, Fuzzy mixer could override any input with an emergency stop. Fuzzy Mixer mixed joystick confidence values and sensor information. Low joystick confidence meant the system needed to avoid obstacles and drive safely in the direction set by the joystick. High confidence in the joystick meant it accurately reflected user wishes and the sensor system had less influence.

Joystick Monitor checked for changes in joystick position and consistency. A steady joystick position indicated a desire to go in a particular direction. A joystick moving randomly indicated an unsure or out of control driver.

Sensor Expert applied knowledge of sensor combinations by creating a grid and made recommendations on courses of action to take a wheelchair away from an object or to prevent collision. Sensor Expert did not consider the wishes of the user.

Doorway extracted information from Sensor Expert. It was an object avoidance program that avoided objects through a "distance function" algorithm. Distance to an object measured by the sensors determined how the powered-wheelchair should react.

Joystick information was combined with sensor information so that:

$$\text{Output(left)} = \text{Input(left)} - \text{F(right)}$$

$$\text{Output(right)} = \text{Input(right)} - \text{F(left)}$$

Where Output was the resultant wheelchair controller voltage, Input was the joystick voltage, and F was the distance function value generated by the sensor system. They were vector quantities, having two values, one for each wheel (left / right).

"Doorway" was effective at turning the powered-wheelchair away from objects, slowing the wheelchair smoothly as it became closer to objects and centralising the wheelchair between objects (such as door frames). Fuzzy Mixer controlled the relationship between the joystick and sensors and apportioned control to joystick or sensors depending on the environment or wishes of the user. Instantaneous relationships could be: (1) all joystick, no sensors, (2) all sensors, no joystick, or (3) in between.

Fuzzy Mixer constantly assessed inputs. Algorithms apportioned control between inputs:

TargetLeft=(((JS0*Aconf[Joysticksector])+((TargetLeft-125)*(255-Aconf[Joysticksector])))/255)+125

TargetRight=(((JS1*Aconf[Joysticksector])+125+((TargetRight-125)*(255-conf[Joysticksector])))/255)

where; TargetLeft/Right = Desired controller voltages; JS0/1 = Actual joystick values; and Aconf[] = Joystick confidence value.

Algorithms used distance functions to create target values for left and right controller voltages. Distance functions were:

TargetLeft = 2.5*Result[1] + 110

TargetRight = 2.5*Result[0] + 110

Where: Result[] = instantaneous range from the sensors. Result[]was scaled and an offset added. This converted sensor data to a form compatible with the target data. To recognise joystick position in order to make an assessment of the wishes of the user, the joystick map was divided into sectors: Forward, Turn right, Turn left, Spin right, Spin left, Stop and Back. Factors to increase joystick confidence (Aconf[]) were: Joystick agrees with sensor system; Joystick held in a steady position; and Joystick position increased against sensor action. Factors to decrease joystick confidence were: Joystick – sensor conflict; and Joystick not held steady.

If the average joystick position was calculated in real time, a smoothed joystick voltage waveform was created. If the instantaneous voltage was rapidly changing, the instantaneous value would usually be substantially different to the average value, so that usually: Actual voltage ≠ Average voltage. This lack of consistency made joystick confidence lower. In other cases, the instantaneous voltage could be similar to the average voltage. This showed a higher level of control for the user or a better understanding of how to drive. In this case, joystick confidence was increased.

A method was needed to assess the wishes and accuracy of the user which allowed the system to monitor the joystick position. Simple averaging was a possibility but an Integration technique was attempted to improve performance.

Sensor Expert applied a set of algorithms to information from sensors. There were seven possible actions:

- "Nothing" carry on under user control,
- "Stop" collision is imminent, stop immediately,
- "Slow" approaching a dangerous situation, slow down,
- "Turn left" a gentle left turn,
- "Spin left" sharp left turn
- "Turn right" a gentle right turn,
- "Spin right" sharp left turn.

A Sensor Expert Rule Set was extracted from the mapping. A two to eight bit Sensor Byte was created from the sensor arrays. Each sensor array had two bits to represent the position (or not) of an object within the array:

0 no detection for this array;
1 detection in "far" element;
2 detection in "middle" element;
3 detection in "near" element.

These numerical operators were used to search Sensor Byte for object configurations so that Sensor Expert could recommend action.

Sensor Expert algorithms were based on recognition of patterns in Sensor Byte.

Distance functions could prevent a wheelchair from passing through a doorway as the sides reached the minimum allowable distance from an object. Distance function algorithms were adjusted to reduce their effect and allow the powered-wheelchair to move close to (and touch) an object to allow wheelchairs to move through doorways.

A simplified Blackboard framework was used as the program structure. The program was easier to control in this structure as the main modules communicated with a blackboard (MainCode) and passed important data to the blackboard. Code was written in C or Assembly Code. The modules are described in [29]. Code was compiled to a single machine level file loaded into micro-controller memory. A modular structure was adopted to simplify program construction and minimise duplication of code.

The final structure was similar to a Blackboard type framework. However the similarities were limited by the size of micro-controller memory of the on-board real time systems which ruled out the creation of complicated structures. The new algorithms made the systems more predictable. If the joystick and the sensor expert were indicating "forward", the system set the trajectory as straight-ahead although the sensors were still interrogated to determine distance from the nearest object. Speed was reduced as the powered-wheelchair became close to an object.

SpinLeft or SpinRight turned the powered-wheelchair. Although controller voltage settings were set to the spin values, the system tended to apply the spin settings for the minimum time required to turn the powered-wheelchair. The powered-wheelchair rarely performed a “spin” manoeuvre in this mode as the system settings would return to “forward“ mode. The application of a spin manoeuvre for a limited time simulated a user moving the joystick completely to one side to execute a turn. Observing users driving a wheelchair and their use of a joystick, it appeared common for the joystick to be moved in exaggerated movements (even to perform gentle manoeuvres).

When a joystick was in a “turn” position, different algorithms were applied to the system, for example an algorithm that prevented the powered-wheelchair from driving quickly into an obstruction during a TurnRight manouvre.

Testing

The new system was initially tested by driving the powered-wheelchair in an uncluttered environment. System response was fast enough for the wheelchair to navigate along a corridor and align with doorways with the joystick in a forward position. The wheelchair path indicated that Sensor Expert was recommending suitable trajectories.

When operating a joystick controlled vehicle, users tended to use large deflections of the joystick. Controller dynamics and powered-wheelchair physical dynamics made large deflections of the joystick suitable for accurate control. Small deflections caused sluggish reactions or inputs were ignored. Large changes in controller input voltages caused smooth changes to be made to the wheelchair trajectory.

Investigation moved on to testing with human volunteers and in more complicated environments. Human users are sophisticated and capable and the intention was not to replace them but to consider ways of assisting.

Powered-wheelchair systems were tested in a laboratory and then in a variety of environments. Wheelchair users quickly learned how the powered-wheelchair responded with the various systems and learned to apply control signals early and to estimate stopping distances. A set of tests were conducted to compare the speed of human driver alone, a human driver with computer assisted operation using the most recently published system and finally using the new expert systems. Tests were observed and the time taken to complete various set courses was recorded for: human drivers by themselves, and then again with the assistance of the most recently published systems, and then with the assistance of the new expert.

Results

The powered-wheelchair successfully negotiated obstacles in various set courses during testing. Assistive computer systems allowed automatic recovery from collisions.

The new expert systems were compared to the most recently published system in [11,12,13] and to a user controlling the robot without the aid of any sensors. The average best time in seconds to complete various courses for users without any sensors to assist were recorded and compared to the most recently published sensor system and the improved system described in this paper. The different courses used for testing became progressively more complicated.

Results from tests using a simple course in the laboratory with one or two obstacles and a constant open floor space with vertical walls around the edges suggested that the new system performed faster (on average) than the most recently published system. That said, the human operators tended to perform faster without the expert systems or the sensor systems to assist them in this simple environment.

Results from tests in a simple corridor with flat surfaces and sloping surfaces bounded with vertical walls and doorways and with two obstacles offset in a staggered formation, suggested the new system performed faster (on average) than the most recently published system but again the human operators tended to perform faster without the expert systems or the sensor systems to assist them in this relatively simple environment. Results from testing in an empty corridor with flat surfaces and sloping surfaces and bounded with vertical walls and doorways were similar and the new expert systems performed faster (on average) than the most recently published system.

Results from testing in a more complicated corridor with doorways and items on the walls (radiators and door surrounds), doorways to pass through and five or more obstacles offset in a staggered formation showed the new expert systems performing faster (on average) than the most recently published system. In this case (as the environment had become more complicated), human operators tended to perform slower without the expert systems or sensor systems.

Graphical results will be presented at the conference as space is limited here.

Discussion

In simple environments, users completed tasks more quickly without any aid from computer and sensor systems. In more complicated environments (complicated corridor and outside), users completed tasks more quickly with the aid of computer and sensor systems. As the environments became more complicated then human operators found it more difficult to judge the trajectory of the powered-wheelchair. The human users often had to slow or stop the wheelchair and reverse it to avoid collision. When environments became more complicated, then human users consistently performed better with assistance. Items on walls (radiators and door surrounds) sometimes slowed wheelchairs as sensors detected them, whereas human users often ignored them. Overall the assisted tasks were performed more quickly.

Different surfaces, slopes and boundaries tended to turn the wheelchairs and sensors became most useful in steering in those cases. The new automated systems managed to consistently correct the trajectory of the wheelchair to a repeatable standard and outperformed the most recently published systems. Some chaotic factors existed, for example, trailing casters could throw the powered-wheelchair off-line. Variation in floor surface, slope or wheel position could affect results. Delays between sensor systems providing feedback information and controllers passing results of that feedback information to powered-wheelchair motors could also cause variations.

Student's t-test was used to compare means of samples in the results. From each sample, the mean $\bar{x}$ was calculated with a measure of dispersion (range of variation) of data around the sample mean (variance S2) and thence the standard deviation (S). Having obtained those values, they were then used to estimate population mean and variance. Each individual set of tests were not necessarily statistically significant but because pairs of tests took place, it was possible to use a paired-samples statistical test. Results were arranged into two sets of replicate data; pairs of results with and without sensor assistance. The paired samples test was used because people (users) were inherently variable. Pairing removed much of that random variability. When results were analysed using a paired-samples statistical test then results were statistically significant. The paired-samples statistical test shows the use without a sensor system and with a sensor system to be significantly different at $p < 0.05$ (95% probability that this result would not occur by chance alone) and the new systems were significantly better than the most recently published systems at $p < 0.05$.

The new system performed every test faster on average than the most recently published systems.

Research is now investigating whether systems on the powered-wheelchair could also usefully be used to monitor a user in terms of driving skill.

More effective control of the powered-wheelchair could be achieved if more information about the environment was available, especially in tight spaces. More control of the power outputs to the motors would be useful. The system needs to take more direct control of the output for fine manoeuvring.

The position of the joystick was the only indication of the intentions of the user. An extension of this work is further analysing user intent from actions exerted on any input device using a Neural Network.

References

1. Stott, and D. Sanders, “New powered wheelchair systems for the rehabilitation of some severely disabled users,” Int Jrnl of Rehab Res vol. 23 (3), pp. 149-153, 2000.

2. D. Sanders, 2Controlling the direction of “walkie” type forklifts and pallet jacks on sloping ground,” Assembly Automation, vol. 28 (4), pp. 317–324, 2008.

3. D.A. Sanders, “Progress in machine intelligence,” Industrial Robot, vol. 35 (6), pp. 485-487, 2008.

4. D.A. Sanders, and I.J. Stott, “A new prototype intelligent mobility system to assist powered wheelchair users,” Industrial Robot, vol. 26(6), pp. 466-475, 1999.

5. D.A. Sanders, and A. Baldwin, “X-by-wire technology”, Total Vehicle Technology: Challenging current thinking, pp 3-12, 2001.

6. D.A. Sanders, and G.E. Tewkesbury, “A pointer device for TFT display screens that determines position by detecting colours on the display using a colour sensor and an Artificial Neural Network,” Displays, vol. 30(2), pp. 84-96, 2009.

7. D.A. Sanders, S. Urwin-Wright, G.E. Tewkesbury, and B. Gremont “Pointer device for thin-film transistor and cathode ray tube computer screens,” Electronics Letters, vol. 41(16), pp. 894-896, 2005.

8. I.J. Stott, and D.A. Sanders, “The use of virtual reality to train powered wheelchair users and test new wheelchair systems,” International Journal of Rehabilitation Research, vol. 23(4), pp. 321-326, 2006.

9. J. Larsson, M. Broxvall, and A. Saffiotti, “Laser-based corridor detection for reactive Navigation,” Industrial Robot: An int’ jnl, vol. 35 (1), pp. 69–79, 2008.

10. M.H.F. Rahiman, Z. Zakaria, R.A. Rahim et al, “Ultrasonic tomography imaging simulation of two-phase homogeneous flow,” Sensor Review, vol. 29 (3), pp. 266-276, 2009.

11. D.A. Sanders, and I.J. Stott IJ, “Analysis of failure rates with a tele-operated mobile robot between a human tele-operator and a human with a sensor system to assist,” Robotica, vol. 30, pp. 973-988, 2012.

12. D.A. Sanders, “Analysis of the effects of time delay on the tele-operation of a mobile robot in various modes of operation,” Industrial Robot, vol. 36(6), pp. 570–584, 2010.

13. D.A. Sanders, “Comparing ability to complete simple tele-operated rescue or maintenance mobile robot tasks with and without a sensor system,” Sensor Review, vol. 30(1), pp. 40-50, 2010.

14. S. Lee, “Use of infrared landmark zones for mobile robot localization,” Industrial Robot 35(2), pp. 153–159, 2008.

15. S. Lee, “Use of infrared light reflecting landmarks for localization,” Industrial Robot: An international journal 36(2), pp. 138-145, 2009.

16. V. Milanes, J.E. Naranjo, C. Gonzalez, et al, “Autonomous vehicle based in cooperative GPS and inertial systems,” Robotica 26, pp 627-633, 2008.

17. R. Bloss, “Latest unmanned vehicle show features both innovative new vehicles and miniaturization,” Industrial Robot: An international journal, vol. 36(1), pp. 13-18, 2009.

18. C. Connolly “Machine vision advances & applications,” Assembly Automation 29(2), pp. 106-111, 2009.

19. D.A. Sanders, “Recognizing shipbuilding parts using artificial neural networks and Fourier descriptors,” Proceedings of Institution of Mechanical Engineers Part B – Journal of Engineering Manufacture, vol. 223(3), pp. 337-342, 2009.

20. D.A. Sanders, G.E. Tewkesbury, D. Ndzi, A. Gegov, B. Gremont, and A. Little, “Improving automatic robotic welding in shipbuilding through the introduction of a corner finding algorithm,” Journal of Marine Science and Technology, vol. 17 (2), pp. 231-238, 2012.

21. D.A. Sanders, “Introducing AI into MEMS can lead us to brain-computer interfaces and super-human intelligence,” Assembly Automation, vol. 29(4), pp. 309-312, 2009.

22. D.A. Sanders, and A. Gegov, “Artificial intelligence tools for use in assembly automation and some examples of recent applications,” Assembly Automation Journal, vol. 33(2), pp. 184–194, 2013.

23. D.A. Sanders, “Force sensing,” Industrial Robot, vol. 34 (4), pp 268-268, 2007.

24. D.A. Sanders, “The modification of pre-planned manipulator paths to improve gross motions associated with the pick & place task,” ROBOTICA 13, pp. 77-85, 1995.

25. G.E. Tewkesbury, and D.A. Sanders, “The use of distributed intelligence within advanced production machinery for design applications, Total Vehicle Technology: Challenging current thinking, pp. 255-262, 2001.

26. Y.C. Chang, and Y. Yamamoto, “On-line path planning strategy integrated with collision and dead-lock avoidance schemes for wheeled mobile robot in indoor environments,” Industrial Robot 35(5), pp. 421–434, 2008.

27. D.A. Sanders, “Real-time geometric modelling using models in an actuator space and Cartesian space. Journal of robotic systems,” vol. 12(1), pp. 19-28, 1995.

28. D.A. Sanders, G. Lambert, J. Graham-Jones, G.E. Tewkesbury, S. Onuh, D. Ndzi, and C. Ross, “A robotic welding system using image processing techniques and a CAD model to provide information to a multi-intelligent decision module”. Assembly Automation, vol. 30(4), pp. 323-332, 2010.

29. D.A. Sanders, G.E. Tewkesbury and D.C. Robinson, “Simple expert systems to improve an ultrasonic sensor-system for a tele-operated mobile-robot,” Sensor Review Journal, vol. 31(3), pp. 246-260, 2011.